Journal of Data Science 22 (3), 376–392
July 2024

DOI: 10.6339/24-JDS1149
Data Science in Action

Evaluation of Text Cluster Naming with Generative Large Language Models

Alexander J. Preiss[1,*], Caren A. Arbeit[1], Anthony Berghammer[1], John Bollenbacher[1], John V. McCarthy[1], Madeline G. Brom[1], Mike Enger[1], Nicholas Rios Villacorta[1], and Shaquavia Straughn[1]

[1]*RTI International, Durham, NC, 27709, United States*

Abstract

Text clustering can streamline many labor-intensive tasks, but it creates a new challenge: efficiently labeling and interpreting the clusters. Generative large language models (LLMs) are a promising option to automate the process of naming text clusters, which could significantly streamline workflows, especially in domains with large datasets and esoteric language. In this study, we assessed the ability of GPT-3.5-turbo to generate names for clusters of texts and compared these to human-generated text cluster names. We clustered two benchmark datasets, each from a specialized domain: research abstracts and clinical patient notes. We generated names for each cluster using four prompting strategies (different ways of including information about the cluster in the prompt used to get LLM responses). For both datasets, the best prompting strategy beat the manual approach across all quality domains. However, name quality varied by prompting strategy and dataset. We conclude that practitioners should consider trying automated cluster naming to avoid bottlenecks or when the scale of the effort is enough to take advantage of the cost savings offered by automation, as detailed in our supplemental blueprint for using LLM cluster naming. However, to get the best performance, it is vital to test a variety of prompting strategies and perform a small test to identify which one performs best on each project's unique data.

Keywords *cluster profiling; large language model; natural language processing; text clustering; topic modeling; unsupervised learning*

1 Introduction

The success of text clustering for a variety of formerly labor-intensive and manual tasks in qualitative research has led to a new challenge: labeling and making sense of those clusters. For example, take open-text survey responses. Researchers often want to identify themes in a set of responses. When the themes of interest are not known a priori, we must use an inductive coding approach. Traditionally, this is a qualitative task requiring that researchers read texts and iteratively develop a set of common themes, tagging texts with themes. For improved reliability, a second or even third researcher will code the text as well. The process requires subject matter expertise and time. Text clustering allows us to automate some of the most time-consuming steps in this process. We can use clustering algorithms to identify natural groupings of texts,

*Corresponding author. Email: apreiss@rti.org.

Received November 30, 2023; Accepted July 20, 2024

assume that clusters represent themes, and separate texts by cluster. However, this still leaves us with the question, "but what are these clusters?"

Manual cluster naming is a common bottleneck in text clustering projects. In the qualitative approach to inductive coding, researchers develop a deep understanding of themes as they go, so they can name and describe each theme. When we automate that step, we need an alternative approach to understand cluster content. Some automated methods exist to extract keywords from clusters, but in practice, these are often insufficient. Most text clustering projects involve the step of a subject matter expert reading the contents of a cluster to assign it a name. This can negate much of the time and cost savings of the clustering approach. In a recent project, several of the authors of this paper clustered a large corpus of specialized texts, resulting in over 2,000 clusters that would have required detailed domain knowledge to interpret. This experience motivated the need for an efficient way to name these clusters.

1.1 Prior Work

The most common approaches to automate the task of naming and describing clusters of texts are keyword extraction methods. Keyword extraction is the process of identifying words or phrases in a text (e.g., paragraph, document) that best represent that text (Rose et al., 2010). Applied to cluster naming, the simplest approach is to extract the words that occur most frequently in each cluster. However, uninformative words can appear frequently (Kaur and Buttar, 2018), and this approach struggles with varying document lengths and cluster sizes. Selecting keywords by the term frequency-inverse document frequency (TF-IDF) metric can improve keyword quality (Ramos, 2003). In the case of cluster naming, cluster-level TF-IDF (c-TF-IDF) computes TF-IDF while treating each cluster as a single document. c-TF-IDF produces an importance score for each word/phrase, which can be used to extract the words/phrases that are most representative of a cluster, relative to other clusters. The Maximal Marginal Relevance (MMR) algorithm can be used to reduce repetitiveness in keyword sets (Carbonell and Goldstein, 1998). Although keyword extraction can produce good results, it has drawbacks. Keyword sets can often be repetitive, even with preprocessing steps like lemmatizing and stopword removal. Also, words and short n-grams often cannot fully represent the depth of a cluster's topic. Finally, keyword sets are more difficult to interpret than plain language.

Methods from the document understanding/summarization literature can also be applied to name clusters of texts. Document summarization produces a shorter chunk of text that summarizes the key information in a longer document (Ma et al., 2021). The output of this process is typically natural language that humans can read and understand. Summarizing a group (e.g., a cluster) of documents, known as multi-document summarization (MDS), adds layers of complexity in combining documents and handling contradictions or redundancy across documents. Combining documents in MDS can be handled through flat concatenation (simply combining within-class documents) or through a more complex concatenation process involving a weighted graph-based approach (Ma et al., 2021). The combination of transfer learning and transformer models allows data scientists and researchers to use pretrained MDS models, thus reducing the need for labeling data (Hosna et al., 2022; Xiao et al., 2022).

Recent advances in generative large language models (LLMs) have dramatically improved performance on document summarization benchmarks (Kamalloo et al., 2023; Zhang et al., 2023a). The size of GPT-like models and the flexibility of their architecture negate the need to explicitly represent redundancy, duplication, and contradiction across documents. These models are also easy and relatively cheap to implement. However, prompt engineering is necessary

to optimize performance on a specific task (Giray, 2023). Generative LLMs are also prone to hallucination, producing results that are inconsistent, contradictory, or incorrect (Zhang et al., 2023b). Perhaps the largest challenge in using generative LLMs to name text clusters is their context window. These models can process a limited amount of input (around 4,000 to 16,000 tokens at the time of this study). This usually precludes passing the full texts of all documents in a cluster to the model.

We believe it is important to evaluate generative LLMs specifically for text cluster naming for three reasons. First, it is important to evaluate models on specific tasks rather than relying on related benchmarks (Bowman and Dahl, 2021). Benchmarks are a useful tool for model comparison, but every real-world applied machine learning task is unique. The only way to know how well a model will perform on a specific task is to test it. Second, document summarization evaluations like the ROUGE metric (Recall-Oriented Understudy for Gisting Evaluation) typically measure performance by comparing the model's outputs to a "gold standard". But a cluster of texts might be named differently but equally well for different purposes or based on different field-specific terminology or natural languages. We believe that qualitative evaluation is important for performance measurement. Third, we wanted to compare several prompting strategies and different ways to pass information from the cluster to the model. We hypothesized that performance could vary greatly depending on the method used to condense a cluster's contents to fit in a model's context window.

2 Methods

This study consisted of three phases: clustering, cluster naming, and cluster name evaluation. In the clustering phase, we selected two benchmark datasets and fit a clustering model to each. We then used the clusters generated by these models for the following phases. In the cluster naming phase, we prompted a generative LLM in various ways to generate names for our clusters. Human annotators also generated cluster names manually. Finally, in the cluster name evaluation phase, human annotators blindly evaluated model-generated and manually generated cluster names on a variety of domains.

2.1 Data

We used two datasets in this work: a public dataset of patient medical reports extracted from PubMed Central articles (Zhao et al., 2023) (henceforth "PMC Patients") and a dataset of abstracts from awards granted in 2022 by the National Science Foundation (henceforth "NSF Abstracts," downloaded via https://www.nsf.gov/awardsearch/). The two datasets differ in their subject matter and size, allowing us to compare the performance and limitations of different prompting strategies in different contexts.

2.2 Clustering

Throughout the clustering phase, we strove to follow an analytical process that we would use on a typical applied text clustering project. This involved many choices: what preprocessing steps to include, what algorithms to use, etc. In applied projects, we make these choices with the goal of maximizing cluster quality. Here, we did the same. See the limitations section for further discussion of the implications of these choices.

Preprocessing steps included deduplication and length filtering. Deduplication is particularly important when using density-based clustering algorithms (like the one used in this project; see below). The high-density areas generated by duplicates can alter the model's perception of density. We also removed near-duplicates, which were common in the NSF Abstracts dataset, where multi-site proposals were often identical except for the substitution of one university or investigator name for another. Rather than define near-duplicates with a specific string similarity threshold, we used the dedupe.io Python package, which uses active learning to train a model to identify near-duplicates.

Length filtering is a common preprocessing step to reduce the tendency of clustering models to separate documents by length rather than by semantic content. We kept documents between the 5th and 95th character count percentile from the NSF Abstracts dataset and documents between the 25th and 75th percentile from the PMC Patients dataset. These values were chosen such that the longest kept document was roughly twice the length of the shortest, a heuristic identified through prior unpublished work.

We used the BERTopic (BERTopic, 2023a) method to cluster the texts in each dataset. BERTopic yields a cluster ID for each text, such that similar texts will be in the same cluster and dissimilar texts will be in different clusters. Comparing clustering models is notoriously difficult, but the authors have found BERTopic to perform outstandingly in a variety of real-world text clustering projects. We used the default SentenceBERT embedding model (Reimers and Gurevych, 2019), Uniform Manifold Approximation (UMAP) (UMAP, 2018) dimension reduction model, and Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) (Hdbscan, 2016) clustering model. For the PMC Patients dataset, we additionally used BERTopic's automatic topic reduction functionality to reduce the number of clusters.

2.3 Cluster Naming

We used OpenAI's GPT-3.5-turbo model, via the OpenAI Application Programming Interface (API), to generate cluster names. Our general strategy involved giving GPT-3.5-turbo information about a cluster's texts and requesting a name that encapsulates the common theme or topic in the texts. We evaluated four variations on this general strategy, which differed in what information we provided to the model about the cluster. The full text of prompts are provided in Supplemental Table 1.

The simplest strategy we tried was the "document-based" strategy, in which we provided GPT-3.5-turbo with the full texts of 20 randomly selected documents from the cluster. To accommodate the longer prompt that this strategy produced, we used the 16k context window version of GPT-3.5-turbo.

We also tested two keyword-based strategies, which provided GPT-3.5-turbo with keywords rather than full texts. These strategies could be useful in scenarios when privacy or security restrictions preclude passing full texts to a third party. They also can be much less expensive than strategies that involve passing full texts, because the OpenAI API (and others) charge by the token.

In the first keyword-based strategy, which we termed "document keywords", we extracted keywords from five randomly selected documents in each cluster using KeyBERT (KeyBERT, 2022). In the second, "cluster keywords" strategy, we extracted a set of keywords associated with the whole cluster, rather than individual documents, using BERTopic (BERTopic, 2023b). For both strategies, we used the MMR algorithm to extract keywords (Carbonell and Goldstein, 1998). Compared to other keyword extraction algorithms, MMR reduces redundancy by

accounting for information novelty. For example, unlike many other algorithms, it would be unlikely to select "cluster", "clustering", and "text clustering" as keywords for this paper. We used an MMR diversity value of 0.3, removed English stopwords using a scikit-learn count vectorizer, and included keywords from one to three n-grams.

Finally, in the "chained resampling" strategy, we performed the document-based naming strategy three times per cluster, using a different sample of documents each time, to obtain three candidate cluster names. We then provided the list of candidate names to GPT-3.5-turbo and asked it to find a consensus name for the cluster. This strategy could increase the likelihood that the sample is representative of the cluster, especially for large clusters. After cluster name evaluation was complete, we identified a bug in the chained resampling strategy for the PMC Patients dataset (we generated the names using titles instead of abstracts). Regenerating names was not feasible at that point, so we omitted chained resampling from results for the PMC Patients dataset.

2.4 Cluster Naming Evaluation

We evaluated cluster names for a sample of 50 clusters per dataset. We chose this as a feasible number given the project's time and budgetary constraints. To assess naming performance across a wide range of cluster sizes, we randomly sampled 10 clusters from each of the top five cluster size deciles for evaluation. We assumed that naming performance would be similar across the fifth to tenth deciles, because these clusters were of relatively similar sizes. Cluster sizes were very right-skewed, so clusters in the fifth decile were about twice the size of clusters in the tenth decile. For comparison, clusters in the top decile were over six times the size of clusters in the fifth decile. The PMC Patients dataset was much larger, and the clustering model identified roughly twice as many clusters (see results). However, we chose to evaluate the same number of clusters from each dataset, because we prioritized equal statistical power across datasets (rather than evaluating more PMC Patients clusters, which would have necessitated evaluating fewer NSF Abstracts clusters).

Two annotators with a background in education research were assigned the NSF Abstracts dataset, and two annotators with a background in health research were assigned the PMC Patients dataset. Each annotator manually generated names for half of the clusters in their assigned dataset, then blindly evaluated the cluster names for the other half of the clusters. The full annotation guide, covering cluster naming and cluster name evaluation, is presented in Supplemental Section 1.

To generate cluster names, annotators were asked to read a sample of 10 documents from each cluster and could choose to read additional documents at their discretion until they were reasonably confident in their perception of each cluster's topic. Although "reasonably confident" is subjective, we chose this criterion because it reflects real-world annotation approaches. Instructing annotators to read a fixed number or proportion of documents is unrealistic, due to variation in cluster size, specificity, and quality. Annotators reported reading an average of 12 to 35 documents per cluster to generate a cluster name.

Annotators then assessed the quality of each cluster name against an evaluation rubric. The cluster names were blinded, so annotators were not privy to whether the names were generated by a human or a model. Annotators only reviewed names for clusters which were named by another annotator (i.e., they never evaluated their own cluster names). In total, annotators reviewed all names, model-generated and manually generated, for all 50 sampled clusters for each dataset.

The cluster name evaluation rubric was based on the document understanding and summarization literature (Dang, 2005; Fabbri et al., 2021; Kryściński et al., 2020). The rubric assessed each cluster name on five domains: (1) fluency (is the quality of the writing good?); (2) consistency (is the name factually aligned with the source documents?); (3) relevance (does the name reflect only important content from the source documents?); (4) completeness (does the name reflect all important content from the source documents?); and (5) overall name quality. These assessments used a five-point Likert scale (very good, good, neutral, poor, very poor). Annotators also rated the quality of each cluster on the same scale. As in the cluster naming step, annotators were presented with a starting sample of 10 documents and could choose to read additional documents until they felt confident in their assessment of each domain.

Finally, in an adjudication step, annotators chose the best name for each cluster, taking all their assessments and any other intangibles into consideration. Annotators were required to choose a single name as best in all cases except for one: when two or more names were identical, and that name was chosen as best, all prompting strategies that generated that name were given credit for the best name.

2.5 Analysis

Likert scale data were quantified on an ordinal scale of 0 to 4, where "Very Poor" was 0 and "Very Good" was 4. Clusters rated "Neutral," "Poor," or "Very Poor" on cluster quality were omitted from all further analyses. We also conducted a sensitivity analysis where low-quality clusters were not omitted from further analyses.

We visualized the quality of the cluster names on our five evaluation domains and the outcome of the adjudication process: the number of times each prompting strategy was deemed to produce the best name for a cluster.

We used statistical tests to assess the likelihood that differences in quality ratings were the result of chance. First, we used Kruskal-Wallis tests to compare all prompting strategies across each domain. The P value for these tests can be interpreted as the probability of observing our results if ratings for all five prompting strategies were drawn from populations with the same distribution. We did not adjust P values for multiple comparisons, so the results should be interpreted conservatively (i.e., the threshold for significance should be stringent). If the P value is low, it is likely that at least one of the groups has a different population distribution. To assess which groups have different population distributions, we used Mann-Whitney pairwise tests. We compared each model prompting strategy to manual naming. We only conducted Mann-Whitney tests for dataset-domain combinations where the Kruskal-Wallis test P value was relatively low.

We also assessed whether name quality varied across clusters of different sizes. We binned clusters into three groups by the number of texts in the cluster: fewer than 100 texts in the cluster, 100 to 500 texts in the cluster, and over 500 texts in the cluster. We visualized overall name quality for each prompting strategy by cluster size bin.

Finally, we assessed whether cluster quality was associated with cluster size. The association between cluster size and cluster quality does not relate directly to the cluster naming experiment. However, it is an informative secondary analysis, because improving cluster quality is also a goal for any analyst seeking to name clusters. We made contingency tables of cluster quality by cluster size bins, then used chi-squared tests of independence to estimate the likelihood that these variables are associated. We then measured the strength of association using Cramer's V, where 0 indicates no association and 1 indicates perfect association.

2.6 Cost Analysis

Cost and time are the principal motivators for automating the cluster naming process. In addition to assessing cluster name quality, we also estimated the cost associated with each prompting strategy. We compared the average cost to generate a single cluster name and the total cost to generate names for each dataset.

To estimate the average number of tokens in the prompt for each prompting strategy, we used tiktoken, a Python package that estimates the number of tokens within a given input string based on OpenAI's embedding model. To estimate the average cost to name a cluster, we multiplied these token counts by the OpenAI API per-token cost. Although we used GPT-3.5-turbo to generate cluster names, we used the per-token cost for OpenAI's latest state-of-the-art model at the time of this analysis (GPT-4-1106-preview, commonly known as GPT-4-turbo) for the cost analysis, because it is a more realistic estimate of the current cost of implementing these prompting strategies. To estimate the total cost to generate cluster names for each dataset, we multiplied the average cost to generate a single name by the number of clusters in each dataset.

To estimate the cost associated with manually naming each cluster, annotators first estimated the time spent per cluster, by referring to timesheets submitted during the naming period. We used the highest and lowest estimates, across the four annotators, as lower and upper bounds. Because the hourly wage for annotators may vary widely depending on context, complexity, and necessary subject matter expertise, we also used lower and upper bounds for this variable. We used \$4/hour as a lower bound to reflect a typical hourly rate for low-cost services like Mechanical Turk. We used \$50/hour as an upper bound to reflect a typical subject matter expert.

3 Results

3.1 Datasets

The NSF Abstracts dataset included abstracts for 7,437 awards given by NSF in 2022. After deduplication, 7,400 documents remained. After filtering to documents between the 5th and 95th percentile in length, 6,661 documents remained. The shortest remaining document contained 1,788 characters and the longest remaining document contained 4,355 characters. The PMC Patients dataset included 167,034 clinical notes. After filtering to documents between the 25th and 75th percentile in length, 83,578 documents remained. The shortest remaining document contained 1,645 characters and the longest remaining document contained 3,531 characters.

3.2 Clustering

For the NSF Abstracts dataset, the clustering procedure yielded 123 clusters ranging in size from 10 to 171 documents. Of the 6,661 documents, 4,607 were assignable to one of the identified clusters while 2,054 were deemed to be outliers – documents that the model judged as not belonging in one of the groups dense enough to call a cluster. For the PMC Patients dataset, the clustering procedure yielded 274 clusters ranging in size from 10 to 4,701 documents. Of the 83,578 documents, 50,467 were clustered and 33,111 were deemed outliers.

3.3 Cluster Name Evaluation

As described in 2.4, 50 clusters from each dataset were selected for evaluation. Of these, ten clusters were omitted from further analysis for having cluster quality ratings of "Poor" or "Very Poor." For the NSF Abstracts dataset, of the 40 remaining clusters, 15 were rated "Very Good" and 25 were rated "Good" on cluster quality. For the PMC Patients dataset, 25 were rated "Very Good" and 15 were rated "Good" on cluster quality. All further analyses assessed the names generated by each prompting strategy for these 40 clusters.

3.3.1 NSF Abstracts

For the NSF Abstracts dataset, model-generated names were competitive with human-generated names across all name quality domains. Overall quality is shown in Figure 1, with findings for the four specific quality domains shown in Supplemental Figures 1 to 4. These plots show Likert scale data. Each plot shows a single domain (e.g., overall quality in Figure 1). Each bar represents the distribution of ratings across the 40 clusters for a single prompting strategy. For example, in Figure 1, of the 40 names generated by chained resampling, 13 were rated very good, 15 were rated good, 9 were rated neutral, and 3 were rated poor.

Supplemental Figure 1 depicts Likert scale ratings for fluency evaluations of 40 text clusters named using four model prompting strategies and manual naming, ratings assigned by human annotators. As shown, fluency evaluations had a different pattern than the other domains (Supplemental Figures 2 to 4). Model-generated names from all prompting strategies were rated higher than human-generated names. Highly fluent responses are easy to produce with the latest generative LLMs. On all other domains, human-generated names were rated second best, and model names generated with the chained resampling prompting strategy were rated best. Keyword-based names tended to be rated worse than other model-generated names, and keyword-based names performed particularly poorly on the completeness domain.

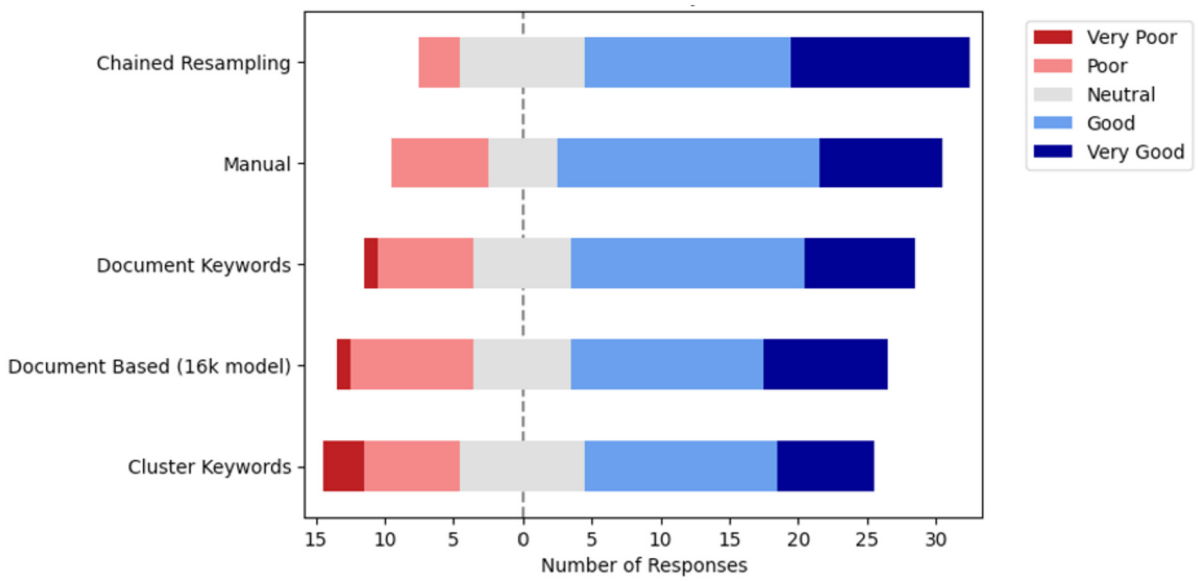

Figure 1: NSF abstracts overall name quality.

Table 1: Illustrative example of NSF abstracts naming results.

Prompting Strategy	Name	Overall Quality Rating
Chained Resampling	Diversity and Inclusion in Geosciences	Very Good
Document Keywords	Geoscience Education and Research	Poor
Document Based (16k model)	Equity and Community Engagement in Geoscience Education	Neutral
Manual	Geoscientist Community Diversification	Neutral
Cluster Keywords	Earth and Environmental Sciences	Poor

An illustrative example from the NSF Abstracts dataset is shown in Table 1. This cluster of abstracts focused on diversity and inclusion (D&I) efforts in the field of geosciences. The chained resampling name captured this essential information, did not include irrelevant information, and used clear language. The keyword-based strategies did not capture the D&I aspect of the cluster, likely because geoscience keywords were selected over D&I keywords by the keyword generation algorithms. The document-based and manual names captured the essential information, but the wording was less clear than the chained resampling name.

The adjudication results for the NSF Abstracts dataset, shown in Figure 2, detail the count of times each prompting strategy was deemed by human annotators to produce the best name for a text cluster and were similar to the overall quality results. Note that the sum of adjudications does not equal 40 because ties were possible when more than one prompting strategy generated identical names. Chained resampling was a clear winner, while other automating prompting strategies were roughly tied with manual naming.

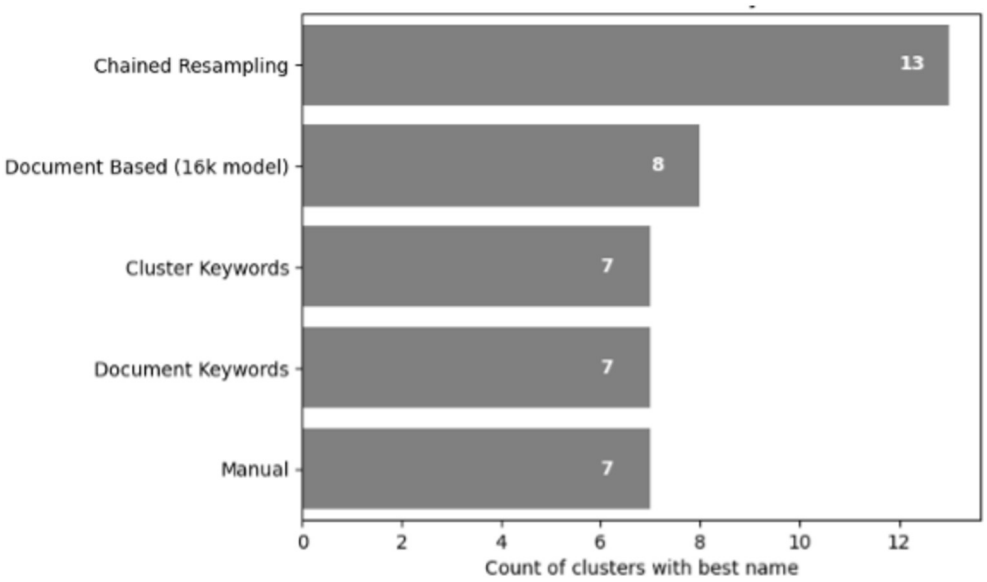

Figure 2: NSF abstracts adjudication results across the 40 text clusters.

3.3.2 PMC Patients

For the PMC Patients dataset, model-generated names were also competitive with human-generated names. As noted in Section 2.3, we omitted chained resampling from the results for the PMC Patients dataset due to a bug in the name generation process. Therefore, we cannot compare its performance across datasets. However, the other strategies performed quite differently on the PMC Patients clusters. Both keyword-based names outperformed document-based names. For the NSF Abstracts clusters, the keyword-based names tended to be too broad. The keyword sets tended to omit aspects of the cluster, which were then omitted from the cluster names. This is likely because of differences in the data that allowed the keyword extraction algorithms to capture more of the PMC Patients clusters' essential elements.

Overall quality ratings of the 40 PMC Patients clusters, by prompting strategy, are shown in Figure 3. Ratings by specific quality domain are shown in Supplemental Figures 5 to 8. As shown in Supplemental Figure 5, human-generated names were rated best for name fluency, but in close competition with the cluster keyword and document-based prompting strategies. Supplemental Figures 6 to 8, however, show that the cluster keyword prompting strategy outperformed the manual approach across the domains of name consistency, relevance, and completeness.

An illustrative example from the PMC Patients dataset is shown in Table 2. The cluster keyword-based name is concise and at the right level of specificity. The manual and document-based names are overly specific. The document keyword-based name is too broad.

The adjudication results, shown in Figure 4, were again similar to the overall quality ratings. Here, the keyword-based prompting strategies were clear winners, while the document-based strategy underperformed relative to manual naming. As with the NSF Abstracts adjudication results shown in Figure 2, the sum does not equal 40 due to ties when more than one prompting strategy generated identical names.

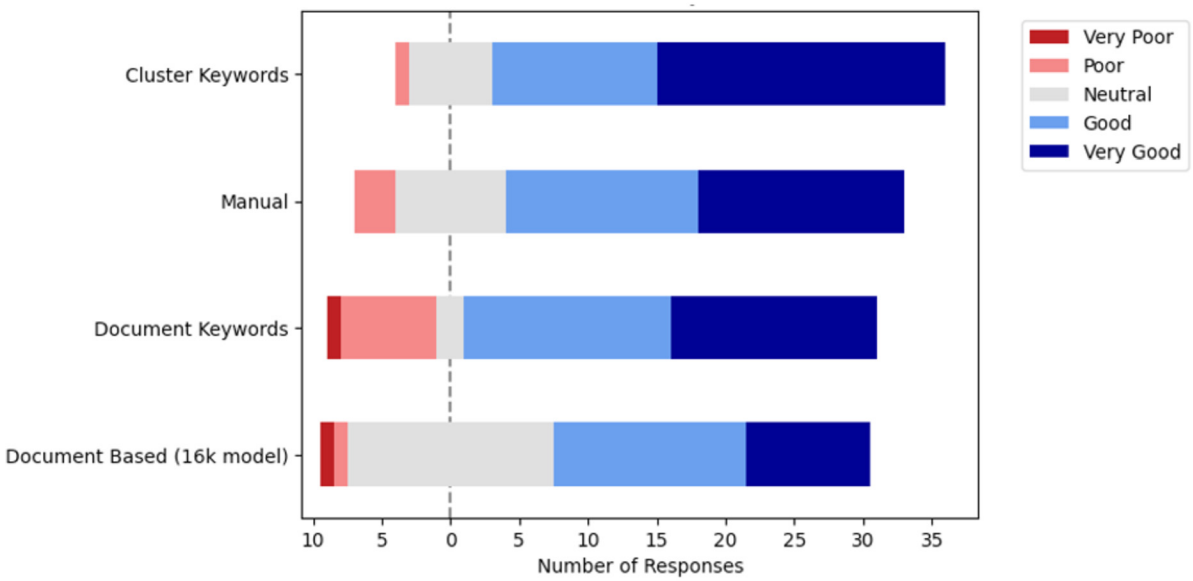

Figure 3: PMC patients overall name quality.

Table 2: Illustrative example of PMC patients naming results.

Prompting Strategy	Name	Overall Quality Rating
Document Keywords	Bleeding disorders	Good
Manual	Platelet disorders requiring intravenous treatment	Good
Cluster Keywords	Platelet disorders	Very Good
Document Based (16k model)	Immune Thrombocytopenia (ITP)	Neutral

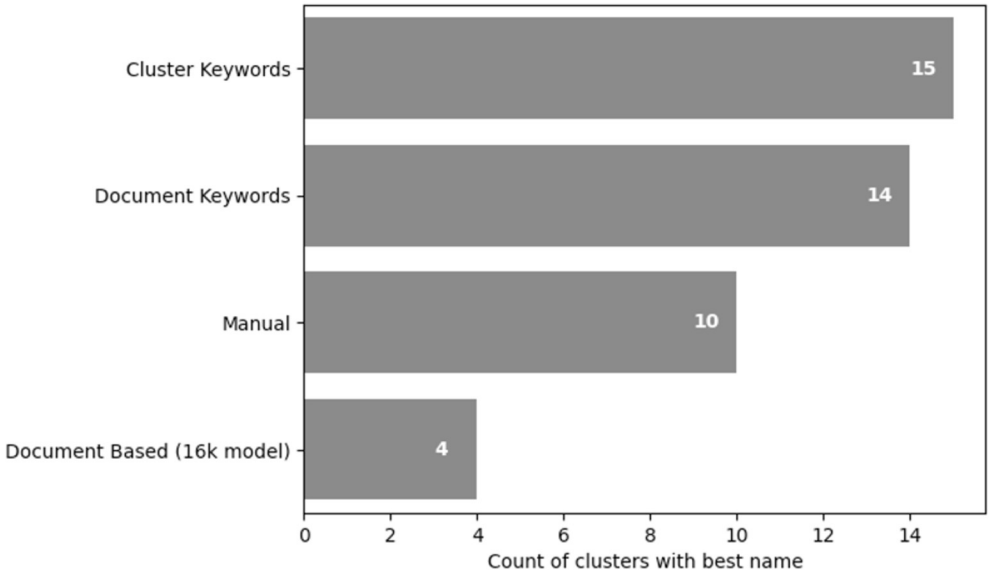

Figure 4: PMC patients adjudication results across the 40 text clusters.

3.3.3 Statistical Tests

We also conducted statistical testing to determine whether the observed differences in quality ratings were statistically significant. First, we used Kruskal-Wallis tests for each quality domain to determine whether any of the prompting strategies significantly outperformed others. Kruskal-Wallis test results are shown in Table 3.

Table 3: Kruskal-Wallis tests for differences in name quality by dataset and domain.

	NSF		PMC	
	Test Statistic	P Value	Test Statistic	P Value
Name Fluency	1.625199	0.804256	0.342806	0.951787
Name Consistency	5.708878	0.221970	5.449346	0.141700
Name Relevance	3.930005	0.415562	3.904997	0.271907
Name Completeness	8.624882	0.071191	15.766742	0.001266
Name Quality Overall	5.585471	0.232318	8.582980	0.035381

Table 4: Mann-Whitney pairwise tests for differences in name quality by prompting strategy (compared to manual naming) and domain.

	Name Completeness		Name Quality Overall	
	Test Statistic	P Value	Test Statistic	P Value
Document Keywords	875.5	0.430383	782.0	0.858792
Document Based (16k model)	547.5	0.011492	651.0	0.133374
Cluster Keywords	901.0	0.287545	942.0	0.144168

Table 5: Cluster count by cluster size bin and dataset.

Cluster Size Bin	Count of NSF Clusters	Count of PMC Clusters
<100	36	15
100–500	4	18
>500	0	7

Overall, the differences in the NSF Abstracts ratings are likely the result of chance. For the PMC Patients dataset, some prompting strategies outperformed others for name completeness and overall name quality. Therefore, we conducted Mann-Whitney pairwise tests to identify which prompting strategies outperformed the manual prompting strategy on these domains (Table 4). The only significant difference was the document-based prompting strategy compared to manual naming on name completeness. These tests demonstrate that prompting strategies were largely indistinguishable from one another in statistical terms.

We also assessed whether overall name quality varied by cluster size. The PMC Patients dataset is much bigger than the NSF Abstracts dataset, so its clusters also tended to be bigger (Table 5). For both datasets, there were no clear differences in overall quality by cluster size (Supplemental Figures 9 to 13). For the NSF Abstracts dataset, model-generated names appeared somewhat better for larger clusters and manual names appeared somewhat worse. However, there were only four clusters in the 100–500 bin, so this was likely the result of chance.

Finally, we assessed whether cluster quality varied by cluster size. For the NSF abstracts dataset, smaller clusters seem to have better quality (Supplemental Table 2). The strength of association (Cramer's V) is moderate, but the chi-squared P value is relatively high, suggesting that the association may be the result of chance. Again, limited statistical power is a likely culprit. For the PMC Patients dataset, smaller clusters also seem to have higher quality (Supplemental Table 3), with a similar strength of association and high P value.

In the sensitivity analysis, where we did not omit clusters with quality ratings of "Neutral", "Poor", or "Very Poor", all analyses used the full sample of 50 clusters. Results did not differ substantively from the primary analysis. For the PMC Patients dataset, the ranking of prompting strategies by overall quality rating and count of best names were identical, as were statistical test results. For the NSF Abstracts dataset, there was some reshuffling in the rankings of the bottom three strategies, but statistical tests still found that the differences between all prompting strategies were not significant. Overall, we conclude that the findings are not sensitive to the exclusion of low-quality clusters.

Cluster names and evaluation results for the 50 sampled clusters from each dataset are shown in Supplemental Table 4.

3.4 Cost Analysis

Cost comparisons for model-generated and human-generated names are shown in Supplemental Table 5. The least expensive prompting strategy used cluster keywords, at $0.08 and $0.21 to name the entire NSF Abstracts and PMC Patients cluster sets, respectively. The most expensive prompting strategy was document-based, at $15.88 for NSF Abstracts and $36.21 for PMC Patients. Annotators' estimates of average time spent naming one cluster ranged from 10 to 25 minutes. The lower bound for manual naming cost, using 10 minutes per cluster and an hourly wage of $4/hour, was $82.00 for NSF Abstracts and $182.67 for PMC Patients. The upper bound for manual naming cost, using 25 minutes per cluster and an hourly wage of $50/hour, was $2,562.50 for NSF Abstracts and $5,708.33 for PMC Patients. In the least favorable comparison (document-based vs. lower bound manual for PMC Patients), automating cluster naming would result in a 5× cost reduction ($182.67 vs. $36.21). In the most favorable comparison (cluster keyword-based vs. upper bound manual for NSF Abstracts) automating cluster naming would result in a 32,031× cost reduction ($2,562.50 vs. $0.08).

4 Discussion

We assessed the ability of GPT-3.5-turbo to generate names for groups of documents created by text clustering. We clustered two benchmark datasets, each focused on a specialized domain: abstracts from awards granted by the NSF and clinical patient notes. We used four prompting strategies (document-based, document keyword-based, cluster keyword-based, and chained resampling) to generate four candidate names for each cluster. Human annotators manually generated names for a sample of clusters. Finally, human annotators blindly evaluated cluster names on five domains: fluency, consistency, relevance, completeness, and overall quality. Annotators also chose the best name for each cluster.

For the NSF Abstracts dataset, annotators rated the names generated by chained resampling higher than manually generated names on all domains. Names generated by chained resampling were also chosen as the best name nearly twice as often as any other prompting strategy. For the PMC Patients dataset, annotators rated the names generated by the cluster keyword-based prompting strategy higher than manually generated names on all domains but one, and both keyword-based strategies were chosen as the best name more often than manually generated names. (As noted in Section 2.3 and Section 3.3, chained resampling was omitted from PMC Patients results due to a bug, so we do not know whether the keyword-based strategies would have outperformed chained resampling on this dataset.) For both datasets, statistical testing showed that prompting strategies were largely indistinguishable from one another in terms of quality.

Based on these findings, we conclude that overall, annotators determined that model-generated cluster names could be as good as or better than cluster names generated by human experts. For both datasets, the best prompting strategy beat the manual approach across all quality domains. However, name quality varied by prompting strategy and dataset. We hypothesize that this is largely driven by variation in the data: certain prompting strategies may work best for data with certain characteristics. The most prominent difference between NSF Abstracts and PMC Patient clusters is breadth. Qualitatively, we believe that PMC Patients clusters tended to cover broader topics for two reasons. First, the dataset was much larger, which led to more and larger clusters. Larger cluster size meant that the document-based prompting strategy included a smaller proportion of documents in the prompt for PMC Patients. Second,

many NSF Abstracts were naturally grouped because they were responses to specific requests for proposals. For example, the "diversity and inclusion in geosciences" cluster highlighted in Table 1 likely includes many proposals responding to a specific request on that topic. Given these differences, we propose further research on the hypothesis that keyword-based cluster naming works better for broader clusters.

We also assessed whether overall name quality varied by cluster size. We found no evidence that certain prompting strategies perform better or worse on larger or smaller clusters. However, our statistical power to detect differences was low. Finally, we assessed whether cluster quality varied by cluster size. We found that smaller clusters had higher quality ratings, but the relationship was not statistically significant. Therefore, the association may be the result of chance, or we may not have the statistical power to measure it confidently. Nevertheless, given this result, it is surprising that cluster size and name quality did not appear to be associated. This also may be because we did not have the statistical power to detect a relationship.

We included two keyword-based prompting strategies in this study because many projects have security or privacy concerns that preclude sending data to a third party. Generating keywords from data and passing those to third-party APIs may be more feasible. These strategies also use much shorter prompts and can therefore be much less expensive. Keyword-based cluster names performed best for one of the datasets, implying that this can be an effective strategy for projects where full texts cannot be passed to third-party APIs or when API costs are an issue. However, because keyword-based names performed worse on the other dataset, projects using keyword-based naming should conduct thorough testing to ensure that name quality is sufficient.

The cost analysis showed that that model-based naming can be orders of magnitude less expensive than manual naming, and that the keyword-based prompting strategies are far less expensive than the document-based and chained resampling strategies. These are rough estimates, and could vary drastically with different data, different annotators, etc. Also, cost estimates for model-based naming did not account for development time. The time needed to implement these techniques may vary. For any individual developer or organization, it will consume the most time the first time around, and subsequent cluster naming efforts will benefit from code reuse and experience. To determine whether automating cluster naming makes sense for a given project, key variables to consider include data size, annotator cost, developer experience, and likelihood that future projects would benefit from reusing automation code. A detailed blueprint for automating cluster naming on your project is presented in Appendix D.

4.1 Limitations

Although our study found evidence that automated cluster naming methods are viable in data science work, several limitations to the present study should be considered. Most significantly, our study was resource constrained, so we only evaluated two small sets of clusters with a small number of annotators. For the same reason, we were unable to assess inter-annotator agreement.

Because of the complexity of the task of cluster naming and the diversity of possible contexts in which it may be applied, these results may not generalize to all cluster naming applications. We also made a series of opinionated choices in our preprocessing and clustering pipeline. These choices impacted the nature of the clusters and therefore the cluster names. Results may have differed had we used different algorithms, preprocessing steps, etc. Because of these limitations to generalizability, we do not interpret our findings to imply that generative LLMs *are* as good as humans at naming text clusters. Instead, we interpret our findings as a demonstration that

generative LLMs *can be* as good as humans at naming text clusters, and we hope it serves to motivate practitioners to try this technique in their projects, even if success is uncertain.

Furthermore, the large variation in the performance of our prompting strategies across the two datasets suggests that in real-world applications, the performance of these prompting strategies needs to be evaluated for each use case, and the best of several strategies selected. This requirement to evaluate prompting strategies in each use case is a significant burden, which may limit the applicability of these methods to the largest scale clustering projects, with thousands of clusters of technical texts.

Additionally, we believe that results may vary significantly by model and prompt. In this work, we evaluated GPT-3.5-turbo, and performed only limited prompt optimizations, leaving a large space of unexplored possible variations. Furthermore, our results were produced with GPT-3.5-turbo during July 2023, and subsequent updates of the model may yield variations in specific prompt performance. For these reasons, application-specific prompt engineering may be required for optimal performance. Nevertheless, we believe the general prompts used here are robust enough to deliver adequate performance in a variety of circumstances, because we did not notice large variations in ad hoc performance assessments during a brief, initial phase of prompt engineering. For projects where maximizing cluster name quality is less important, a general approach, such as the generative cluster naming now built in to BERTopic, may be adequate.

We note that the document keyword-based and chained resampling approaches may have been handicapped by our decision to sample five documents per cluster. We initially sampled five documents for the document-based approach as well, and used the same number across prompting strategies for consistency. The upgrade to 20 documents was a late-breaking response to the availability of the 16k context window version of GPT-3.5-turbo. We were unable to update the document keyword-based and chained resampling approaches prior to the start of cluster name evaluation. This may have particularly impacted the document keyword-based approach, because document keyword sets were much shorter than documents, and this approach could plausibly have sampled orders of magnitude more documents. In addition, the need to omit chained resampling from the PMC Patients results due to a naming bug precludes us from determining whether it would have also performed best on another dataset.

Finally, LLM advances since the time of this study have two important implications. First, the latest generation of LLMs has longer context windows, so document-based cluster naming could include more documents in the prompt. This could improve performance relative to other prompting strategies, but it would also be more expensive. Second, at the time of publication, the latest open-source LLMs now outperform the closed-source model used in this study. The gap between open- and closed-source models is getting smaller, and new tools make it possible to run open-source LLMs on consumer or enterprise infrastructure. Overall, open-source models have become a far more feasible option for real-world generative tasks. Using these models does not require sending prompt data to a third party, so the privacy and security concerns that motivated the keyword-based prompts may be less relevant in the future.

4.2 Conclusion

Naming and describing clusters of texts is a common bottleneck in text clustering projects. In this study, we tested four approaches to name text clusters using generative LLMs. We compared model-generated names to names generated by human experts through a rigorous evaluation process. Overall, we demonstrated that model-generated cluster names can be as good as or better than cluster names generated by human experts. We conclude that text clustering

practitioners should consider trying automated cluster naming when cluster naming presents a bottleneck or when the scale of the effort is enough to take advantage of the cost savings offered by automated cluster naming. However, to get the best performance, it is vital to test a variety of prompting strategies and perform a small test to identify which one performs best on each project's unique data.

Supplementary Material

Appendices A–D.

Acknowledgement

The authors would like to thank the creators of the datasets used in this work. We thank the United States National Science Foundation for providing bulk award download functionality, as well as the investigators who wrote the award abstracts. We thank Zhengyun Zhao, Qiao Jin, Fangyuan Chen, Tuorui Peng, and Sheng Yu for compiling the PMC-Patients dataset and making it publicly available. We also thank the authors of the many PubMed Central articles included in that dataset. Finally, we would like to acknowledge several RTI colleagues who supported the internal funding mechanism that made this work possible (Shellery Ebron, Lauren Zitney, and Edward Preble) and who provided valuable feedback on this study's approach and execution (Peter Baumgartner).

Funding

This work was funded internally by an RTI International research and development funding mechanism.

References

BERTopic (2023a). The algorithm. Accessed 2023.

BERTopic (2023b). c-tf-idf. Accessed 2023.

Bowman SR, Dahl GE (2021). What will it take to fix benchmarking in natural language understanding? arXiv preprint: https://arxiv.org/abs/2104.02145.

Carbonell J, Goldstein J (1998). The use of mmr, diversity-based reranking for reordering documents and producing summaries. In: *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 335–336.

Dang HT (2005). Overview of DUC 2005. *Technical report*, National Institute of Standards and Technology (NIST).

Fabbri AR, Kryściński W, McCann B, Xiong C, Socher R, Radev D (2021). Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9: 391–409. https://doi.org/10.1162/tacl_a_00373

Giray L (2023). Prompt engineering with ChatGPT: A guide for academic writers. *Annals of Biomedical Engineering*, 51: 2629–2633. https://doi.org/10.1007/s10439-023-03272-4

Hdbscan (2016). The hdbscan clustering library. Accessed 2023.

Hosna A, Merry E, Gyalmo J, Alom Z, Aung Z, Abdul M (2022). Transfer learning: A friendly introduction. *Journal of Big Data*, 9: 102. https://doi.org/10.1186/s40537-022-00652-w

Kamalloo E, Dziri N, Clarke CLA, Rafiei D (2023). Evaluating open-domain question answering in the era of large language models. arXiv preprint: https://arxiv.org/abs/2305.06984.

Kaur J, Buttar PK (2018). A systematic review on stopword removal algorithms. *International Journal of Future Revolution in Computer Science & Communication Engineering*, 4(4): 207–210.

KeyBERT (2022). About the project. Accessed 2023.

Kryściński W, McCann B, Xiong C, Socher R (2020). Evaluating the factual consistency of abstractive text summarization. arXiv preprint: https://arxiv.org/abs/1910.12840.

Ma C, Zhang WE, Guo M, Wang H, Sheng QZ (2021). Multi-document summarization via deep learning techniques: A survey. arXiv preprint: https://arxiv.org/abs/2011.04843.

Ramos J (2003). Using TF-IDF to determine word relevance in document queries. *Technical report*.

Reimers N, Gurevych I (2019). Sentence-BERT: Sentence embeddings using siamese BERT-networks. arXiv preprint: https://arxiv.org/abs/1908.10084.

Rose S, Engel D, Cramer N, Cowley W (2010). Automatic keyword extraction from individual documents. In: *Text Mining: Applications and Theory* (MW Berry, J Kogan, eds.). John Wiley & Sons, Ltd.

UMAP (2018). UMAP: Uniform manifold approximation and projection for dimension reduction. Accessed 2023.

Xiao W, Beltagy I, Carenini G, Cohan A (2022). Primera: Pyramid-based masked sentence pre-training for multi-document summarization. arXiv preprint: https://arxiv.org/abs/2110.08499.

Zhang T, Ladhak F, Durmus E, Liang P, McKeown K, Hashimoto TB (2023a). Benchmarking large language models for news summarization. arXiv preprint: https://arxiv.org/abs/2301.13848.

Zhang Y, Li Y, Cui L, Cai D, Liu L, Fu T, et al. (2023b). Siren's song in the ai ocean: A survey on hallucination in large language models. arXiv preprint: https://arxiv.org/abs/2309.01219.

Zhao Z, Jin Q, Chen F, Peng T, Yu S (2023). PMC-patients: A large-scale dataset of patient summaries and relations for benchmarking retrieval-based clinical decision support systems. arXiv preprint: https://arxiv.org/abs/2202.13876.